third edition

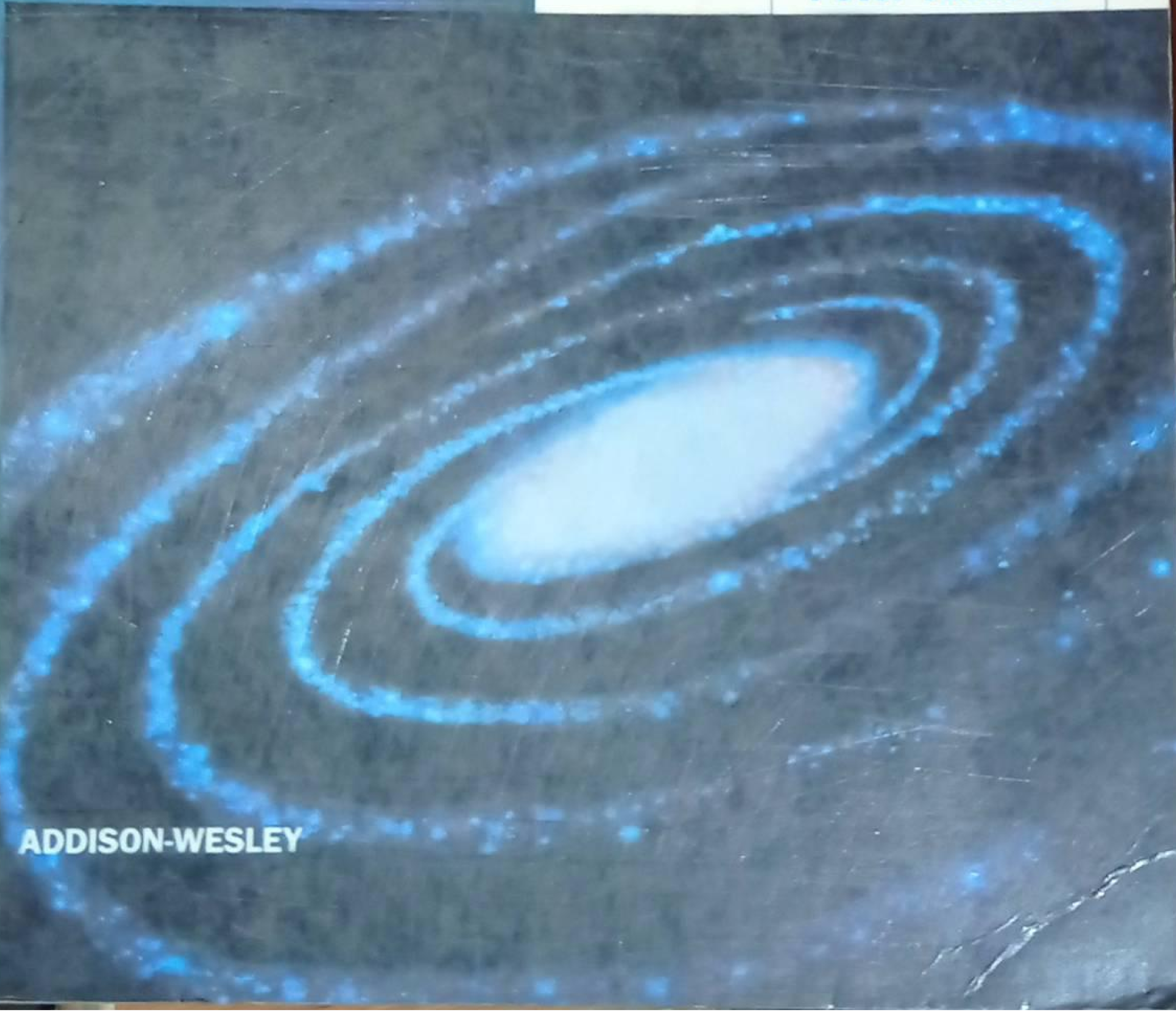
Pearson Education Asia

introduction to

EXPERT SYSTEMS

Pearson Education Asia

Peter Jackson



ADDISON-WESLEY

Scanned by TapScanner

Contents

Preface

1 What Are Expert Systems? 1

1.1 The nature of expertise 1
1.2 The characteristics of an expert system 3
1.3 Fundamental topics in expert systems 4
1.3.1 Acquiring knowledge 4
1.3.2 Representing knowledge 6
1.3.3 Controlling reasoning 8
1.3.4 Explaining solutions 9
1.4 Summary and chapter plan 11
1.4.1 What is the state of the art? 11
1.4.2 Chapter plan of the book 12
Bibliographical notes 13
Study suggestions 14

2 An Overview of Artificial Intelligence 15

2.1 The Classical Period: game playing and theorem proving 16
2.1.1 State space research 16
2.1.2 Heuristic search 21
2.2 The Romantic Period: computer understanding 23
2.2.1 SHRDLU 23
2.2.2 Knowledge representation schemes 25
2.3 The Modern Period: techniques and applications 27
2.3.1 Knowledge is power 28
2.3.2 'AI winter' and 'AI spring' 31
Bibliographical notes 32
Study suggestions 32

3 Knowledge Representation 38

3.1 The representation of knowledge: principles and techniques 38
3.2 The STRIPS planner 41
3.2.1 Operator tables and means–ends analysis 41
3.2.2 Assessment of STRIPS representation and control 45
3.3 Subgoaling in MYCIN 46
3.3.1 Treating blood infections 46
3.3.2 MYCIN's knowledge base 48
3.3.3 MYCIN's control structure 50

Scanned by TapScanner

3.4 Evaluating and comparing expert systems 53
3.4.1 Evaluation of MYCIN 53
3.4.2 Comparison with STRIPS 55
Bibliographical notes 57
Study suggestions 57

4 Symbolic Computation 60

4.1 Symbolic representation 61
4.2 Physical symbol systems 61
4.3 Implementing symbol structures in LISP 62
4.3.1 LISP data structures 63
4.3.2 LISP programs 64
4.3.3 Functional application and lambda conversion 66
4.3.4 List processing 67
4.3.5 Pattern matching 69
4.4 Why LISP isn't a knowledge representation language 71
4.4.1 Symbol level versus knowledge level 71
4.4.2 LISP and program design 71
4.5 Languages for knowledge representation 72
Bibliographical notes 74
Study suggestions 74

5 Rule-Based Systems 76

5.1 Canonical systems 77
5.2 Production systems for problem solving 78
5.2.1 The syntax of rules 78
5.2.2 The working memory 81
5.3 Controlling the behavior of the interpreter 85
5.3.1 Conflict resolution 85
5.3.2 Forward and backward chaining 87
5.3.3 Rules and meta-rules 93
Bibliographical notes 96
Study suggestions 96

6 Associative Nets and Frame Systems 100

6.1 Graphs, trees and networks 100
6.2 The rise of associative networks 104
6.2.1 The type–token distinction and cognitive economy 104
6.2.2 Assessing the adequacy of associative nets 105
6.3 Representing typical objects and situations 107
6.3.1 Introduction to frame concepts 107
6.3.2 Complex nodes in a network 108
6.3.3 Defaults and demons 109
6.3.4 Multiple inheritance and ambiguity 112
6.3.5 Comparing nets and frames 115

Scanned by TapScanner

Bibliographical notes 116
Study suggestions 116

7 Object-Oriented Programming 120

7.1 Prototypes, perspectives and procedural attachment 121
7.2 LOOPS and Flavors 123
7.2.1 Message passing 123
7.2.2 The method combination problem 125
7.2.3 Metaclasses 127
7.3 CLIPS and the Common LISP Object System (CLOS) 129
7.3.1 Multiple inheritance in CLOS and CLIPS 129
7.3.2 Method combination in CLOS and CLIPS 131
7.3.3 Metaclasses in CLOS and CLIPS 133
7.4 Multiple inheritance in C++ 135
7.5 Object-oriented analysis and design for expert systems 140
Bibliographical notes 141
Study suggestions 141

8 Logic Programming 143

8.1 Formal languages 143
8.1.1 Propositional calculus 144
8.1.2 The predicate calculus 145
8.2 The PROLOG language 148
8.3 Resolution refutation 149
8.3.1 The resolution principle 149
8.3.2 Proof search in resolution systems 151
8.4 Procedural deduction in PLANNER 154
8.5 PROLOG and MBASE 156
8.5.1 PROLOG's search rules 156
8.5.2 Explicit search control in MBASE 157
Bibliographical notes 162
Study suggestions 162

9 Representing Uncertainty 166

9.1 Sources of uncertainty 166
9.2 Expert systems and probability theory 168
9.2.1 Conditional probability 168
9.2.2 Certainty factors 170
9.2.3 Certainty factors versus conditional probabilities 172
9.3 Vagueness and possibility 175
9.3.1 Fuzzy sets 175
9.3.2 Fuzzy logic 177
9.3.3 Possibility theory 178
9.4 The uncertain state of uncertainty 179
Bibliographical notes 180
Study suggestions 181

Scanned by TapScanner

10 Knowledge Acquisition 182

10.1 Theoretical analyses of knowledge acquisition 183
10.1.1 Stages of knowledge acquisition 183
10.1.2 Different levels in the analysis of knowledge 184
10.1.3 Ontological analysis 187
10.2 Expert system shells 187
10.2.1 EMYCIN as architecture and abstraction 188
10.2.2 Maintaining and debugging knowledge bases in TEIRESIAS 189
10.3 Knowledge acquisition methods 191
10.3.1 Knowledge elicitation by interview in COMPASS 191
10.3.2 Automating knowledge elicitation in OPAL 193
10.3.3 A graphical interface to a domain model 194
10.3.4 Efficacy of OPAL and related efforts 197
10.4 Knowledge-based knowledge acquisition 197
Bibliographical notes 199
Study suggestions 199

11 Heuristic Classification (I) 207

11.1 Classifications of expert system tasks 208
11.2 Classification problem solving 210
11.2.1 Heuristic matching 210
11.2.2 The generality of heuristic classification 212
11.3 Classification versus construction 215
Bibliographical notes 218
Study suggestions 218

12 Heuristic Classification (II) 225

12.1 Mapping tools to tasks 225
12.2 Heuristic classification in MUD and MORE 227
12.2.1 A model of the drilling fluid domain 227
12.2.2 Knowledge acquisition strategies 231
12.2.3 Confidence factors in MORE 234
12.2.4 Evaluating MORE 235
12.3 Making strategy more explicit 236
12.3.1 Lessons of the GUIDON project 236
12.3.2 NEOMYCIN's task structure 238
Bibliographical notes 240
Study suggestions 240

13 Hierarchical Hypothesize and Test 243

13.1 Managing complexity 244
13.2 Structured objects in CENTAUR 245
13.2.1 The structure of prototypes 246
13.2.2 Rules embedded in prototypes 247
13.3 Model-based reasoning in INTERNIST 248

Scanned by TapScanner

13.3.1 Representing knowledge in a disease tree 248
13.3.2 Focusing attention in INTERNIST 251
13.3.3 Practical and theoretical problems with INTERNIST 252
13.4 TDE as knowledge engineering workbench 253
Bibliographical notes 255
Study suggestions 256

14 Constructive Problem Solving (I) 259

14.1 Motivation and overview 259
14.2 A case study: R1/XCON 260
14.2.1 Components and constraints 261
14.2.2 Using contexts to impose task structure 263
14.2.3 Reasoning with constraints: the Match method 265
14.3 Elicitation, evaluation and extensibility 267
14.3.1 Knowledge elicitation in R1/XCON 268
14.3.2 The evaluation and extension of R1/XCON 270
Bibliographical notes 272
Study suggestions 272

15 Constructive Problem Solving (II) 275

15.1 Construction strategies 275
15.2 An architecture for planning and meta-planning 277
15.3 Eliciting, representing and applying design knowledge 284
15.3.1 Knowledge-based backtracking in VT 284
15.3.2 Acquiring propose and revise knowledge in SALT 286
15.4 Summary of constructive problem solving 288
Bibliographical notes 290
Study suggestions 290

16 Designing for Explanation 294

16.1 Rule-based explanation 294
16.1.1 MYCIN's explanation system 295
16.1.2 Explanation in MYCIN derivatives: EMYCIN and NEOMYCIN 297
16.2 Frame-based explanation 299
16.2.1 Explanation in CENTAUR 299
16.2.2 Multimedia interfaces for explanation 306
16.3 Explanation and automatic programming 308
16.3.1 Automatic programming in XPLAIN 308
16.3.2 The Explainable Expert Systems project 309
16.3.3 Text plans and user models in PEA 312
16.4 Explanation facilities and future research 314
Bibliographical notes 315
Study suggestions 315

17 Tools for Building Expert Systems 320

17.1 Overview of expert systems tools 320

Scanned by TapScanner

17.2 Expert system shells 322
17.2.1 Matching shells to tasks 322
17.2.2 Shells and inflexibility 323
17.3 High-level programming languages 324
17.3.1 Constraints of production rule languages 324
17.3.2 Evaluating object-oriented approaches 325
17.3.3 Logic programming for expert systems 326
17.3.4 Multiple-paradigm programming environments 328
17.3.5 Additional modules 330
17.4 Potential implementation problems 333
17.4.1 Common pitfalls and how to avoid them 333
17.4.2 Selecting a software tool 334
17.4.3 How easy is it to use these tools? 336
17.4.4 What is good programming style? 338
17.5 More maxims on expert system development 339
Bibliographical notes 341
Study suggestions 341

18 Blackboard Architectures 344

18.1 The blackboard metaphor 344
18.2 HEARSAY, AGE and OPM 346
18.2.1 Motivation for HEARSAY-II architecture 346
18.2.2 HEARSAY's use of knowledge sources 347
18.2.3 HEARSAY-III: an abstract architecture 348
18.2.4 Abstraction in AGE and OPM 349
18.3 The blackboard environment BB* 350
18.3.1 Architecture, framework and application 351
18.3.2 BB1 and ACCORD as architecture and framework 351
18.3.3 PROTEAN: application as framework instantiation 352
18.3.4 Integrating different reasoning strategies 354
18.3.5 Summarizing BB* 355
18.4 Efficiency and flexibility in blackboard frameworks: GBB and Erasmus 356
18.4.1 Blackboard retrieval in GBB 356
18.4.2 Blackboard configuration in Erasmus 357
18.5 Concurrency and parallelism in CAGE and POLIGON 358
Bibliographical notes 360
Study suggestions 360

19 Truth Maintenance Systems 362

19.1 Keeping track of dependencies 362
19.1.1 Relaxation in networks 363
19.1.2 Belief revision 364
19.2 Revising propositional theories 366
10.3 Nonmonotonic justifications 367
19.4 Maintaining multiple contexts 371
19.4.1 Assumption-based maintenance 371

19.4.2 Model-based diagnosis using ATMS 374
19.5 Summary and comparison of TMSs 377
Bibliographical notes 378
Study suggestions 378

20 Machine Learning 380

20.1 Overview of inductive learning 381
20.2 Early work: Meta-DENDRAL 383
20.2.1 Rule generation and refinement 384
20.2.2 Version spaces 386
20.2.3 The candidate elimination algorithm 387
20.2.4 Matching instances to patterns in Meta-DENDRAL 388
20.3 Building decision trees and production rules 390
20.3.1 The structure of decision trees 390
20.3.2 The ID3 algorithm 392
20.3.3 Changes and additions to ID3 in C4.5 395
20.4 Tuning rule sets 397
Bibliographical notes 399
Study suggestions 400

21 Belief Networks 402

21.1 Dempster–Shafer theory 402
21.1.1 Belief functions 403
21.1.2 Applying Dempster–Shafer theory to MYCIN 404
21.2 Pearl's theory of evidential reasoning in a hierarchy 406
21.3 Comparing methods of inexact reasoning 408
21.4 Summarizing the state of uncertainty 410
Bibliographical notes 411
Study suggestions 411

22 Case-Based Reasoning 413

22.1 The case base 414
22.1.1 The CHEF program 415
22.1.2 Retrieval and adaptation methods 416
22.2 Computer-aided instruction: the CATO system 418
22.2.1 The domain of caselaw 418
22.2.2 Legal research and legal reasoning 419
22.2.3 CATO as an intelligent teaching system 420
22.3 Case-based report generation in FRANK 422
22.3.1 Components of FRANK 423
22.3.2 FRANK's blackboard system 424
22.3.3 The CBR module of FRANK 424
22.4 Comparing case-based and rule-based systems 425
Bibliographical notes 427
Study suggestions 427

Scanned by TapScanner

23 Hybrid Systems 430

23.1 Learning methods in ODYSSEUS 431
23.1.1 EBG as abstraction 431
23.1.2 Case-based learning 433
23.2 The ODYSSEUS and MINERVA systems 434
23.2.1 The MINERVA expert system shell 434
23.2.2 Learning in ODYSSEUS 435
23.3 Using cases to handle exceptions 437
23.4 Hybrid systems and connectionist approaches 439
23.4.1 Why connectionism? 439
23.4.2 SCALIR: a hybrid system for legal information retrieval 443
23.5 Learning in SCALIR 446
23.6 The future of hybrid systems 447
Bibliographical notes 448
Study suggestions 448

24 Summary and Conclusion 450

24.1 The riddle of artificial intelligence 450
24.2 Knowledge representation revisited 452
24.3 AI programming languages 454
24.4 Practical problem solving 455
24.5 Expert system architectures 457
24.6 Expert systems research 458
24.7 Conclusion 459
Bibliographical notes 460
Study suggestions 461

Appendix CLIPS Programming 462

A1 A short history of CLIPS 462
A2 Rules and functions in CLIPS 463
A2.1 Facts 463
A2.2 Rules 465
A2.3 Watching and dribbling 467
A2.4 Using templates 468
A2.5 Defining functions 468
A3 Object-orientation in CLIPS 469
A4 The 'Knights and Knaves' example 474
A4.1 Understanding the problem 474
A4.2 Ontological analysis and knowledge representation 476
A4.3 Writing the rules 478
A4.4 Extending the rules: logical compounds 481
A4.5 Backtracking and multiple contexts 488
A4.6 Handling reported speech 498
A4.7 Complete program listing 501

Scanned by TapScanner

A5 CLIPS programming style 512
Study suggestions 513

References 514

Index 532

Scanned by TapScanner